Better Practices for Domain Adaptation

Anonymous ICCV submission

Paper ID ****

Abstract

Distribution shifts are all too common in real-world applications of deep learning. Domain adaptation (DA) aims to address this by providing various frameworks for adapting models to the deployment data without using labels. However, the domain shift scenario raises a second more subtle challenge: the difficulty of performing hyperparameter optimisation (HPO) for these adaptation algorithms without access to a labelled validation set. The unclear validation protocol for DA has led to bad practices in the literature, such as performing HPO using the target test labels when, in real-world scenarios, they are not available. This has led to over-optimism about DA research progress compared to reality. In this paper, we devise a more rigorous framework for future work, by benchmarking a suite of candidate validation criteria and using them to assess popular adaptation algorithms. We show that there is a challenge across all three branches of the domain adaptation literature including Unsupervised Domain Adaptation (UDA), Source-Free Adaptation (SFDA), and Test Time Adaptation (TTA). In each case, there is a large gap between oracle HPO and achievable performance given real computable validators. Additionally we highlight the importance of using proper validation splits in order to reliably estimate target generalisation performance. Finally, we find previously unexplored validation metrics that are widely applicable across all settings. Altogether, our improved practices covering data, training, validation and hyperparameter optimisation form a new rigorous pipeline to improve benchmarking, and hence research progress, within the field going forward.

1. Introduction

Supervised deep learning models achieve impressive results when training and testing data are identically distributed. However, perhaps the main failure mode of computer vision and pattern recognition systems in practice is due to the near-ubiquitous distribution shift between data curated for model training, and real-world data encountered during deployment [1]. This distribution shift issue has motivated a tremendous amount of work in the area of unsupervised domain adaptation (UDA) [1]. UDA methods aim to alleviate domain shift by collecting freely available unlabelled data during deployment to a target domain and adapting vision models based on this unlabelled data.

Hundreds of unsupervised adaptation algorithms have now been proposed based on various principles from distribution alignment [10], to domain adversarial learning [4] and much more. However, without exception, a key challenge for every one of these algorithms is *how to tune hyperparameters and conduct model selection?* In conventional supervised learning, hyperparameters and model selection (stopping criteria) are handled systematically by maximising accuracy on a validation split of the training set. In unsupervised domain adaptation there is no such straightforward solution because the target domain has no labels with which to compute accuracy, and the source domain is not representative of the target domain.

Despite the importance of this issue—upon which any practical application of domain adaptation hinges—there has been relatively little systematic study of validation protocols and algorithms for UDA [29, 2, 18]. Worse, a recent meta-review and re-evaluation of the domain adaptation literature found that most published code did not use consistent or fair model selection criteria [12], and furthermore when evaluated under consistent and fair model selection criteria most existing results can not be replicated [12]. This mini "replication crisis" in domain adaptation highlights the need for studying validation protocols for UDA, and for fair benchmarking to drive reliable progress.

The few existing fair model selection criteria for UDA are based on diverse intuitions such as simply applying UDA algorithm objectives on the validation split of the unlabelled target set, priors on the expected distribution of labels [2, 18], or relying on the validation accuracy in the source domain [29]. However there is little first principles justification to pick among these reasonable intuitions, and there is little empirical evaluation to understand which are best, and how close they come to the performance of an oracle validator, which has been the basis of many reported results in the literature [12].

These challenges exist throughout the domain adaptation literature. They arise across all three popular branches of adaptation for recognition: Unsupervised Domain Adaptation (UDA) [4, 21, 23], Source-Free Domain Adaptation (SFDA) [8, 28] and Test Time Adaptation (TTA) [26, 30]. They also arise across different kinds of learning problems from classification [?] to regression [?], dense prediction [?], and detection [?].

To address this issue, we conduct a large-scale benchmark of 5 domain adaptation algorithms with 9 different validation criteria and three DA settings (UDA, SFDA, TTA). We identify which DA validators can be applied to each setting, and characterise the size of the challenge in each case in terms of the gap between practically achievable and best-case DA performance. We identify effective practices in terms of using validation splits to estimate target performance. Finally, we report which are the best existing validators. This data point should drive future practice both in DA research – which should use these validators, rather than unrealistic oracle HPO; and in validator research – which should aim to develop validators which surpass the best that we report.

2. Related Work

2.1. Domain Adaptation

There are now too many domain adaptation algorithms to review here, and we refer the reader to good surveys such as [1, 13]. Most deep UDA algorithms proceed by performing supervised learning on the source domain data, and some kind of unsupervised objective on the target domain data. Representative families of approach include objectives that penalise misalignment between the source and target domain feature distributions [10], train a domain classifier that can then be used adversarially to penalise distinguishable source and target domain features [4], or penalise deviation from a prior on the expected target label distribution [20]. However, all algorithms have a number of hyper-parameters, such as stopping iteration and strength of the weighting factor for supervised vs unsupervised loss components. How to set these hyper-parameters is not clear given the lack of a labelled target domain validation set in UDA applications.

The long-established mainstream setting for unsupervised domain adaptation (UDA) assumes that source and target data are accessed simultaneously for training. Two related problem variants have more recently gained rapid popularity, namely source-free domain adaptation (SFDA) and Test Time Adaptation (TTA). SFDA refers to the condition where pre-trained source models should be adapted to the target data without revisiting the source data [8] – for example, by unsupervised fine-tuning. TTA [25, 22] similarly adapts a pre-trained model without access to the source data, but assumes that the test data arrives in mini-batches, providing the opportunity to adapt to each mini-batch before making decisions on their labels. Both of these settings obviously still have many hyperparameters (e.g., learning rate, number of iterations, regularisation strengths) for the proposed algorithms, and hence suffer from the lack of a clear validation protocol in a DA context. None of the seminal studies in this area show valid HPO criteria in their papers or code.

2.2. Validation Approaches for DA

Comparatively few papers have systematically studied validation criteria for UDA, given the importance of this issue for practical application of UDA. Typical solutions applied by UDA algorithm papers include: (1) Oracle risk. Many papers use the target test set for hyperparameter selection [12], which is obviously incorrect as it can not be used in real applications; (2) Source risk. Evaluating the adapted model on the source validation set is reasonable but may not be a good validation criterion due to domain shift between source and target domains; (3) Evaluating another UDA algorithm objective (such as InfoMax [20] and MMD [10]) on an unlabelled validation split of the target set; (4) Validation domain. Use of a held-out labelled validation domain, as used in the VisDA challenge [14], is fair. However, this assumes multiple labelled domains, which may not be available in practice, and also raises additional questions of whether the optimal hyperparameters for the validation domain are representative of the optimal hyperparameters for the target domain.

Besides the above strategies, a few purpose-designed validation criteria have been proposed: Deep embedding validation (DEV) [29] weights the source validation risk by the probability that each sample belongs to the source domains. Meanwhile, Silhouette Score [15], Batch nuclear-norm minimisation (BNM) [2], and soft neighbourhood density (SND) [18] criteria that boil down to evaluating the adapted models' posterior label distribution on the target domain under different notions of a prior for the expected target domain label distribution. Mean ensemble-based validation (ENS) [15] considers a linear combination of the above criteria in an attempt to improve performance. However, overall it is unclear which to prefer for UDA.

2.3. Benchmarking Domain Adaptation

There have been two major benchmarking exercises in UDA. The VisDA competition challenge [14] provides a labelled validation domain for model selection and hyper-parameter optimisation (HPO). However, validation domains may not be available in practice—-and if they are, they may not be representative of the target domain. Thus, the vast majority of research literature on UDA has not used this approach. A recent empirical evaluation [12, 11] analysed the GitHub repositories of a number of UDA methods and found that: (1) In practice different methods used very different validation criteria for empirical evaluation, making published results incomparable with each other; (2) A large number of prior studies used the oracle risk as a validation criterion, meaning that their results are not representative of how well domain adaptation would work in reality using validation criteria that can be implemented in practice; (3) Variation in existing validation criteria was high compared to variation across adaptation algorithms, and none of them were strongly correlated with recognition performance. Our evaluation extends this early study but goes beyond it in considering all three major branches of DA research (UDA, SFDA, TTA), considering a wider variety of validators, and demonstrating how validator performance can be improved through proper use of validation splits within the target domain.

3. Background

3.1. Problem Setup

Unsupervised Domain Adaptation: In the UDA setup, one typically trains a model $f_{\boldsymbol{\theta}} : \mathcal{X} \mapsto \mathcal{Y}$ on a labelled dataset, $\mathcal{D}_S = \{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N_S}$, consisting of data sampled from a source domain, p_S. The goal is then to adapt $f_{\boldsymbol{\theta}}$ using an unlabelled dataset, $\mathcal{D}_T = \{\boldsymbol{x}_i\}_{i=1}^{N_T}$, sampled from a target domain, p_T. The general learning objective to be minimised w.r.t. to $\boldsymbol{\theta}$ can be simplified as follows,

$$L(f_{\boldsymbol{\theta}}, \mathcal{D}_S, \mathcal{D}_T) = L_{\text{sup}}(f_{\boldsymbol{\theta}}, \mathcal{D}_S) + L_{\text{da}}(f_{\boldsymbol{\theta}}, \mathcal{D}_S, \mathcal{D}_T), \quad (1)$$

where $L_{\text{sup}}(\cdot)$ is typically the cross-entropy loss for classification and mean square error for regression problems, and $L_{\text{da}}(\cdot)$ is the adaptation loss, such as MMD [23], CORAL [21] and DANN [4] losses,

Source-Free Domain Adaptation: The SFDA setting aims to adapt a pre-trained source domain model to the target domain, relaxing the assumption of joint occurrence of source and target domain data in UDA. So first, a source model will be optimized using source domain data

$$\hat{\theta} = \arg\min_{\theta} L_{\text{sup}}(f_{\boldsymbol{\theta}}, \mathcal{D}_S) \quad (2)$$

then trained source model $\hat{\theta}$ will be adapted to the target domain by

$$\theta^* = \arg\min_{\hat{\theta}} L_{\text{sfda}}(f_{\hat{\theta}}, \mathcal{D}_T). \quad (3)$$

where now L_{sfda} is typically an unsupervised loss, such as pseudo labelling [?], information maximization [?] and clustering [?] losses.

Test-Time Adaptation: Unlike SFDA, TTA assumes the batch-wise target domain data $X \sim \mathcal{D}_T$ comes in a stream and adapts a pre-trained source model for each minibatch X as

$$\theta^* = \arg\min_{\hat{\theta}} \underset{X \sim \mathcal{D}_T}{L_{\text{tta}}} (f_{\hat{\theta}}, X), \quad (4)$$

where L_{tta} is commonly the unsupervised loss, such as self-supervised learning and entropy minimisation losses.

3.2. Model Selection

The *de facto* model selection is that the best candidate model configured by a hyperparameter set ($\boldsymbol{h} \in \mathbb{H}$, $\mathbb{H}$ is the pool of hyperparameter sets) will be selected based on their evaluation score, $d(f_{\boldsymbol{\theta}}, \mathcal{D}_V)$[1], where $\mathcal{D}_V$ is a validation dataset. The process can be formalised as

$$\begin{aligned} \boldsymbol{h}^* &= \arg\max_{\boldsymbol{h}} \ d(f_{\boldsymbol{\theta}^*_{\boldsymbol{h}}}, \mathcal{D}_V), \\ \text{s.t. } \boldsymbol{\theta}^*_{\boldsymbol{h}} &= \arg\min_{\boldsymbol{\theta}} \ L(f_{\boldsymbol{\theta}}, \mathcal{D}_S, \mathcal{D}_T; \boldsymbol{h}). \end{aligned} \quad (5)$$

However, two things in UDA complicate this process: 1) determining how to select the validation set $\mathcal{D}_V$; and 2) deciding which evaluation metric should be used if $\mathcal{D}_V = \{\boldsymbol{x}_i\}_{i=1}^{N_V}$ is an unlabelled set from the target domain.

A recent work systematically investigated the possible validation criteria for UDA, which we summarise below using $\hat{\boldsymbol{y}}$ to denote the one-hot predictions of the model and $\boldsymbol{y}$ as the one-hot ground truth labels.

Source accuracy: d is simply the accuracy metric and $\mathcal{D}_V$ can be a training or validation set from a source domain.

$$d(f_{\boldsymbol{\theta}}, \mathcal{D}_V) = \frac{1}{N_V} \sum_{i=1}^{N_V} \mathbf{1}(\hat{\boldsymbol{y}} = \boldsymbol{y}), \quad (6)$$

where $\mathbf{1}(\cdot)$ is the indicator function that evaluates to one if its argument is true and zero otherwise.

Entropy: Entropy has been used in an adaptation loss [26] as well as for model selection. In this case, d computes the confidence of the model predictions, as measured by the entropy of the predicted label distribution, and $\mathcal{D}_V$ is typically the training or validation set from an unlabelled target domain. We further investigate the effect when $\mathcal{D}_V$ comes from the source domain.

$$d(f_{\boldsymbol{\theta}}, \mathcal{D}_V) = \frac{1}{N_V} \sum_{i=1}^{N_V} H(\boldsymbol{p}_i), \boldsymbol{p}_i = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \quad (7)$$

where

$$H(\boldsymbol{p}) = -\sum_{j=1}^{K} p_{[j]} \text{log} p_{[j]}, \quad (8)$$

computes the entropy of the categorical distribution, $\boldsymbol{p}$.

[1] Assuming the model performance is a monotonically decreasing function of the output of $d(\cdot, \mathcal{D}_V)$.

Information maximisation (IM): IM is often used as an adaptation loss as well [20] to maximise the diversity of prediction in addition to confidence.

$$d(f_{\boldsymbol{\theta}}, \mathcal{D}_V) = H(\frac{1}{N_V}\sum_{i=1}^{N_V} \boldsymbol{p}_i) - \frac{1}{N_v}\sum_{i=1}^{N_v} H(\boldsymbol{p}_i). \tag{9}$$

Adjusted Mutual Information (AMI): This is the adjusted mutual information between predicted and cluster labels.

$$d(f_{\boldsymbol{\theta}}, \mathcal{D}_V) = \mathrm{AMI}(\boldsymbol{p}, \mathrm{CL}(\mathcal{D}_V)) \tag{10}$$

where $\mathrm{CL}(\mathcal{D}_V)$ is the cluster labels for validation set $\mathcal{D}_V$, which can be the target training or validation set.

V-Measure: Similarly to AMI, this is a metric defined over clustering labels and predictions. It is defined as the harmonic mean between homogeneity and completeness [16].

Other clustering measures: Along with AMI and V-Measure, we compute several other related clustering measures, namely, adjusted Rand index, Fowlkes–Mallows index, silhouette score, Davies–Bouldin index and Calinski-Harabasz index.

RankMe: Originally proposed for estimating the transferability of self-supervised representations [5], RankMe approximates the rank of the feature matrix on pre-training data. We investigate its application to both source and target domain data.

CORAL: CORAL is an adaptation algorithm that aligns the feature distributions of the source and target data by minimising second-order statistics [21]. Their loss can be used as a validator and is defined as the difference between the covariance matrices of the two domains, C_S and C_T.

$$\begin{aligned} d(f_{\boldsymbol{\theta}}, \mathcal{D}_V) &= \mathrm{CORAL}(\mathcal{D}_S, \mathcal{D}_T) \\ &= \frac{1}{4d^2}\|C_S - C_T\|_F^2 \end{aligned} \tag{11}$$

Maximum mean discrepancy (MMD): A common metric used to compute the discrepancy of feature distributions from source and target domains [23], which can be used with the assumption that the trained model may have a good target performance when the source and target domain features are aligned.

$$\begin{aligned} d(f_{\boldsymbol{\theta}}, \mathcal{D}_V) &= \mathrm{MMD}(\mathcal{D}_S, \mathcal{D}_T) \\ &= \frac{1}{N_S(N_S-1)}\sum_{i=1}^{N_S}\sum_{j\neq i}^{N_S} k(vs._{i}, vs._{j}; f_{\boldsymbol{\theta}}) \\ &+ \frac{1}{N_T(N_T-1)}\sum_{i=1}^{N_T}\sum_{j\neq i}^{N_T} k(\boldsymbol{t}_i, \boldsymbol{t}_j) \\ &- \frac{2}{N_S N_T}\sum_{i=1}^{N_S}\sum_{j=1}^{N_T} k(vs._{i}, \boldsymbol{t}_j), \\ k(a,b) &= \exp\left\{\frac{-\|a-b\|_2^2}{e}\right\}, \end{aligned} \tag{12}$$

where $vs.$ and $\boldsymbol{t}$ are the features extracted for the data from source and target domains, respectively. And additionally, the **Class-wise MMD** which computes the MMD distance between source and target domains from the same classes separately.

$$\begin{aligned} d(f_{\boldsymbol{\theta}}, \mathcal{D}_V) &= \text{CW-MMD}(\mathcal{D}_S, \mathcal{D}_T) \\ &= \frac{1}{C}\sum_{i=1}^{C} \mathrm{MMD}(\mathcal{D}_S|_{\boldsymbol{y}=i}, \mathcal{D}_T|_{\boldsymbol{y}=i}) \end{aligned} \tag{13}$$

When MMD is used for validation, the validation set combines the train sets or validation sets of source and target domains.

Soft neighbourhood density (SND): SND computes the entropy based on the gram matrix of the validation features.

$$\begin{aligned} d(f_{\boldsymbol{\theta}}, \mathcal{D}_V) &= H(\alpha(\boldsymbol{X}, \tau)), \\ \boldsymbol{X} &= \boldsymbol{v}^T\boldsymbol{v}, \end{aligned} \tag{14}$$

where $\boldsymbol{v}$ are the data features, $\alpha(.)$ and τ are softmax function and temperature. Here $\mathcal{D}_V$ can be the train or validation set of source or target domains.

Batch nuclear-norm maximization (BNM): BNM was originally a UDA algorithm, which maximizes the nuclear norm of the prediction matrix in a batch, being repurposed as a validation criterion.

$$d(f_{\boldsymbol{\theta}}, \mathcal{D}_V) = \|\boldsymbol{P}\|_*, \boldsymbol{P} = f_{\boldsymbol{\theta}}(\mathcal{D}_V) \tag{15}$$

where $\boldsymbol{P} \in R^{N_V \times C}$ the prediction matrix of whole data in $\mathcal{D}_V$ using $f_{\boldsymbol{\theta}}$. And $\|\|_*$ computes the nuclear norm.

4. Evaluation

Our evaluation extends the benchmark of [12]. We make their setup more rigorous by splitting the target domain into

Table 1. How the source and target domains are split and how each split is used for (1) the source-only model and (2) adaptation algorithms.

	Source		Target		
	Train	Val	Train	Val	Test
Source-only	Train	Validate	-	-	Test
Adaptors	Adapt	-	Adapt	Validate	Test

Table 2. Summary of adaptation algorithms considered

	Algorithm	Approach
UDA	ATDOC [9]	Pseudo-labelling
	BNM [3]	SVD loss
	DANN [4]	Adversarial
	MCC [7]	Information maximisation
	MCD [19]	Classifier discrepancy
	MMD [10]	Feature distance
SFDA	AAD [28]	Clustering
	NRC [27]	Graph clustering
	SHOT [8]	Information maximisation
TTA	TENT [25]	Entropy minimisation
	TTT++ [22]	Self-supervised learning

train/val/test sets. Previous works often compute target performance on the same data that the algorithms adapt to, or the same data that the validators use. This fails to properly measure generalisation performance as we will show later. Ours splits and how we use them are detailed in Table 1.

In order to compare different validation criteria, we collect a large set of model checkpoints that span several datasets, algorithms, feature layers and hyperparameter choices. We want the optimal validator to behave similarly to the target domain test performance of the corresponding algorithm. We measure the quality of each validator in two ways: 1) computing the Spearman rank correlation between validator scores and oracle test accuracy, 2) using the validator to select the best model for an algorithm/task pair and comparing the test performance of it against the best model as selected by the oracle.

4.1. Unsupervised Domain Adaptation

4.1.1 Setup

Datasets: We gather model checkpoints from a wide range of UDA benchmark datasets: MNIST-M [4] which consists of one setup from standard MNIST to a modified version; VisDA2017-C which contains *train*, *validation* and *test* domains — we consider the shifts *train* → *validation* and *train* → *test*; Office31 [17] which consists of three domains: *amazon*, *dslr* and *webcam*; and OfficeHome [24] with four domains: *art*, *clipart*, *product* and *real*.

Algorithms: We consider six representative domain adaptation algorithms, spanning both recent and classic methods and a variety of underlying principles. These include the pseudo-label based **ATDOC** [9]; domain-adversarial learning with the seminal **DANN** [4]; domain-alignment with **MMD** [10]; **BNM** and **MCC** which optimise the target label distribution under nuclear norm prior and minimum class confusion priors respectively, and the classifier-discrepancy-based **MCD** [19].

Table 3. Summary of validation criteria considered

Criterion	Approach
BNM [3]	Label prior
AMI [11, 15]	Label prior
ARI	Label prior
V-Measure	Label prior
FMI	Label prior
Silhouette [11, 15]	Label prior
DBI	Label prior
CHI	Label prior
MMD [10]	Domain Alignment
CORAL [21]	Domain Alignment
SND [18]	Label prior
InfoMax [20]	Label prior
Entropy	Label prior
Source Accuracy	

We start by finetuning a network on the source task. We take ResNet50 weights pretrained on ImageNet [6] for all datasets apart from MNIST-M where a smaller CNN is used. The final classification layer is replaced by an MLP head consisting of two blocks of {Linear, ReLU, Dropout} followed by a final Linear layer. We finetune only this head on the source task using a standard categorical cross-entropy loss. 10 models are trained with learning rates sampled uniformly at random from a logarithmic scale between $10^{-5} - 10^{-1}$. These runs form the set of checkpoints for the *source-only* model. They are not used for evaluating the validation criteria, but we report performances at times for comparison.

We use a diverse set of state-of-the-art adaptation algorithms (see Table 2) to collect our checkpoints. When training each adaptation algorithm, we use the source-only model weights as initialisation for both the backbone and MLP head. The specific source-only checkpoint used as initialisation is the one with the highest source validation accuracy and in case of ties we select the checkpoint trained for the fewest amount of epochs.

We generate a set of feature checkpoints using 6 algorithms, 20 datasets, 10 hyperparameter samples, 2 feature layers, and record 20 different checkpoints during training. This gives us a total of 48,000 checkpoints.

Validators: We compare our two new validators to a large number of existing criteria, listed in Table 3.

Questions: Using the setup above, we aim to answer the following questions: (i) How well do the various validation criteria correlate with true testing performance? (ii) Which validation criterion leads to the best generalisation perfor-

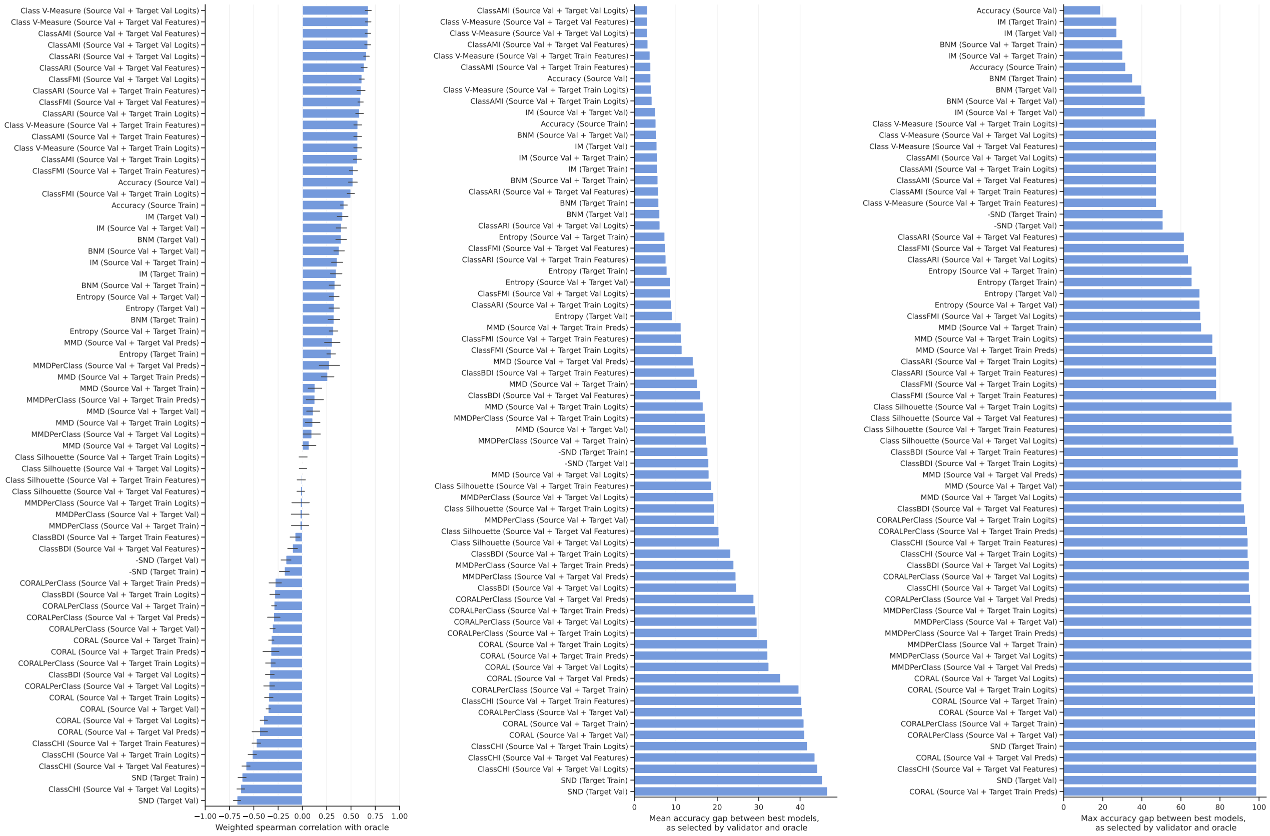

Figure 1. Left: Correlations with target test accuracy on UDA benchmarks (individual domain plots in **??**). Error bars are standard error across domains. Middle: Average gap between the best model as selected by each validator and the oracle. Right: Maximum gap between the best model as selected by each validator and the oracle.

mance when used for model selection? (iii) What is the impact of validating on the training set versus an independent validation split?

4.1.2 How well do validation criteria correlate with testing performance?

Using our 48,000 model checkpoints, we compute the true target domain testing performance for each; as well as each checkpoint's score under the various validation criteria. The results, aggregated across all pairs of source and target domains in the Office31 dataset, are shown in Figure 1; and broken down across each pair of source and target domains in Figure **??**.

4.1.3 Which validation criterion leads to the best generalisation performance when used for model selection?

We next use the various scores to perform model selection and hyperparameter optimisation for each of the base DA algorithms.

4.1.4 What is the impact of validating on training vs validation splits?

As discussed in [12, 11], while prior work that validates on the source domain has fairly consistently used the source validation set; prior work that validates on the unlabelled target domain has been inconsistent with regard to choice of validation on the target train set or an independent target val set. Our normative theory for validation requires validation on the target val set. However, because prior criteria are often intuitively motivated there is also not an obvious answer about which is preferred. To analyse this issue, we compare using the train vs validation split for evaluating criteria. From the results in Table 5 we see that for almost all the criteria the val split is preferred. While this result might seem unsurprising in retrospect, we emphasise that the use of a val split is NOT standard practice in the literature, even in thorough recent evaluations [11].

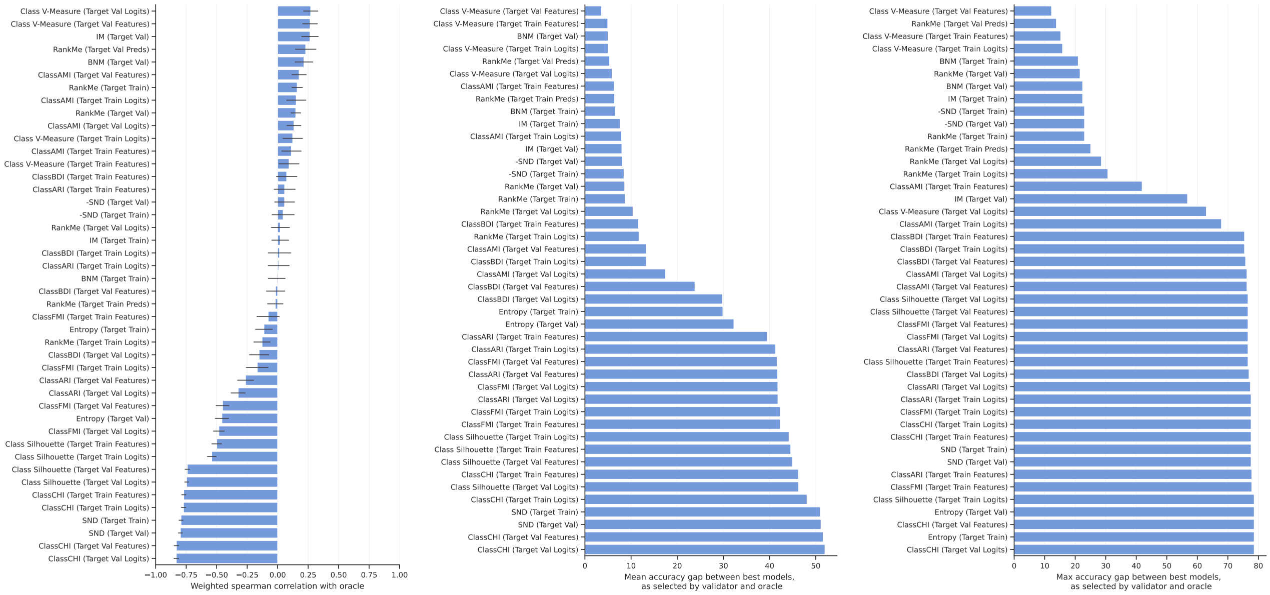

Figure 2. Left: Correlations with target test accuracy on SFDA benchmarks (individual domain plots in **??**). Error bars are standard error across domains. Middle: Average gap between the best model as selected by each validator and the oracle. Right: Maximum gap between the best model as selected by each validator and the oracle.

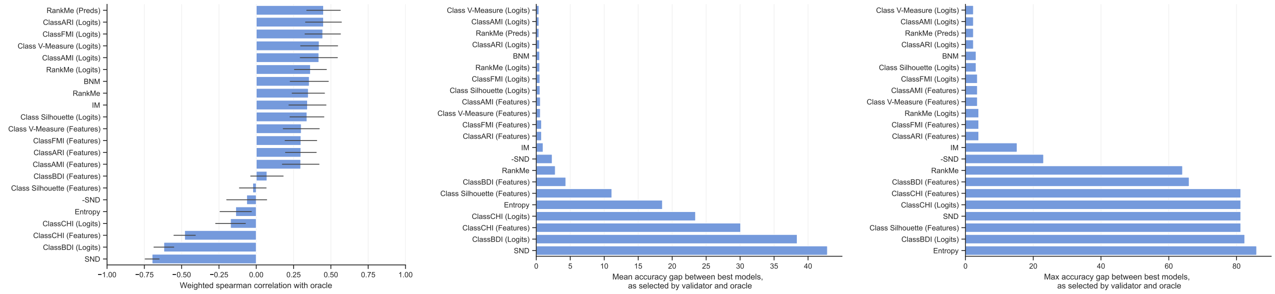

Figure 3. Left: Correlations with target test accuracy on TTA benchmarks (individual domain plots in **??**). Error bars are standard error across domains. Middle: Average gap between the best model as selected by each validator and the oracle. Right: Maximum gap between the best model as selected by each validator and the oracle.

4.2. Source-free Domain Adapation

4.2.1 Setup

For SFDA we use the OfficeHome dataset as a benchmark, covering all 12 domain shifts. The same source-only models that we produced for UDA are also used here for initialisation of the same architecture. Three recent SFDA algorithms adapt the model on target domain data, AAD [28], NRC [27] and SHOT [8]. For each algorithm, we sample 10 sets of hyperparameters and train for 200 epochs

Validators: As the source domain is not available in this setting, we can only apply our validators to the target domain splits. Following our results in Section 4.1.4 we use the target validation split for all validators. **Questions:**

4.3. Test-Time Adaptation

4.3.1 Setup

We adopt the TTA setting, where a pre-trained model adapts on the test data as it comes, one batch at a time. In particular, we use the episodic setting where the model is reset after each batch. **Datasets:** We use the most common TTA benchmark of CIFAR-10-C, consisting of 15 versions of the CIFAR-10 test set with different corruptions applied, including Gaussian noise, pixelation and fog. **Algorithms:** We use the pre-trained CIFAR10 checkpoint of [30] as our source-only model and initialisation for the TTA algorithms. TENT [26] adapts by minimising the entropy on its predictions on the test batch, and TTT++ [30] uses a self-supervised auxiliary loss in addition to feature alignment via MMD and CORAL.

Table 4. Comparison of validation criteria for model selection in UDA. We report the target test performance for the top models selected by each validator.

	RankMe	AMI	ARI	V-Measure	FMI	Silhouette	DBI	CHI	BNM	MMD	CORAL	SND	IM	Entropy	Accuracy	Oracle
ATDOC	61.52	68.06	67.67	68.05	67.75	44.96	57.14	16.29	64.83	55.00	18.99	15.81	65.32	60.52	68.06	72.24
BNM	64.32	69.17	69.41	69.17	69.35	61.51	58.27	32.56	64.87	59.17	37.66	27.47	64.82	64.84	66.02	71.09
DANN	63.21	64.48	63.69	64.61	63.69	56.66	56.31	39.50	64.22	57.59	42.17	34.60	63.92	63.68	62.44	68.27
MCC	61.54	69.28	70.12	69.48	70.06	59.43	53.99	24.78	68.29	54.55	23.43	18.98	68.38	60.23	69.11	72.41
MCD	64.44	60.48	48.35	60.66	41.83	16.17	39.86	8.99	64.04	35.20	15.29	8.97	64.16	54.32	63.83	67.75
MMD	60.20	65.33	63.44	65.33	60.94	54.67	58.24	35.72	61.69	56.52	35.73	32.60	62.34	62.29	63.83	67.44
Avg.	62.54	66.13	63.78	66.22	62.27	48.90	53.97	26.31	64.66	53.01	28.88	23.07	64.82	60.98	65.55	69.87
Avg. Rank	7.17	3.17	4.08	2.83	5.08	11.17	11.00	13.83	5.50	10.83	13.17	15.00	5.17	7.50	4.50	-
Souce-only																65.60

Table 5. Comparison of split for evaluation of validation criteria. We report the average target test accuracy of selected models for each validator when applied on (1) target train data and (2) target validation data.

	RankMe	AMI	ARI	V-Measure	FMI	Silhouette	DBI	CHI	BNM	MMD	CORAL	SND	IM	Entropy	Accuracy
Train	61.77	65.43	62.15	65.59	58.27	50.56	55.23	29.19	64.32	54.76	28.97	24.12	64.53	62.14	64.50
Val	62.54	66.13	63.78	66.22	62.27	48.90	53.97	26.31	64.66	53.01	28.88	23.07	64.82	60.98	65.55

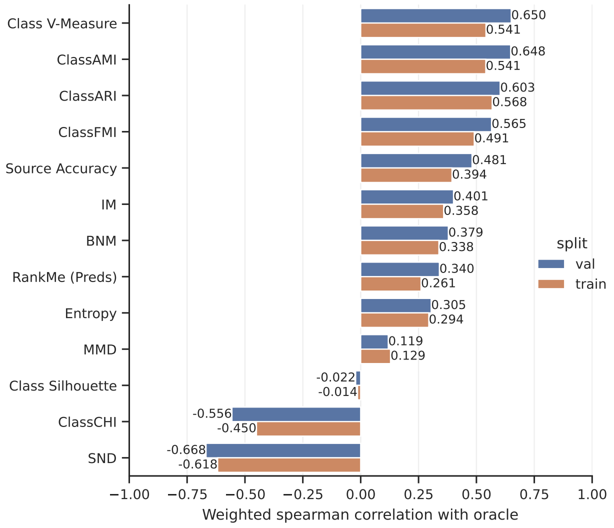

Figure 4. Weighted Spearman rank correlation. Comparison of split for evaluation of validation criteria. We report the average weighted Spearman rank correlation between each validator and target test accuracy when using the following data splits for computing validators: (1) target train data and (2) target validation data. (G-Score uses source train/validation data and for those validators that use both source and target, we always use the validation split in the source domain.)

Validators: As this setting only exposes a single batch to the model at a time, both training and validation use the same data. **Questions:** **Results:** In this setting, the top validators manage to almost match the oracle performance, indicating that ... However, it is worth noting that this is an artificially constructed benchmark. It is likely that on a more realistic dataset like OfficeHome, the trends would be closer to that of SFDA. We leave this investigation for future work.

5. Conclusion

We investigated the problem of model selection criteria for unsupervised domain adaptation. Taking a normative approach based on a target domain generalisation bound, we derived two new principled model selection criteria. Our exhaustive empirical evaluation showed that our criteria both have the strongest correlation to the final testing performance, and are also the most effective for maximising performance when used for model selection. Uniquely, our criteria are general bounds that can be instantiated for different kinds of inference problems, unlike prior work that is restricted to classification. We showed successful instantiations for both classification and regression problems. In future work, we will instantiate them for structured prediction problems such as semantic segmentation.

References

[1] Gabriela Csurka, Timothy M Hospedales, Mathieu Salzmann, and Tatiana Tommasi. Visual domain adaptation in the deep learning era. *Synthesis Lectures on Computer Vision*, 11(1):1–190, 2022. 1, 2

[2] Shuhao Cui, Shuhui Wang, Junbao Zhuo, Liang Li, Qingming Huang, and Qi Tian. Towards discriminability and diversity: Batch nuclear-norm maximization under label insufficient situations. In *CVPR*, 2020. 1, 2

[3] Shuhao Cui, Shuhui Wang, Junbao Zhuo, Liang Li, Qingming Huang, and Qi Tian. Towards Discriminability and Diversity: Batch Nuclear-norm Maximization under Label Insufficient Situations. In *CVPR*, 2020. 5

[4] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. 2016. 1, 2, 3, 5

[5] Quentin Garrido, Randall Balestriero, Laurent Najman, and Yann Lecun. Rankme: Assessing the downstream performance of pretrained self-supervised representations by their rank, 2022. 4

Table 6. SFDA performance on Office-Home.

	RankMe	AMI	ARI	V-Measure	FMI	Silhouette	DBI	CHI	BNM	SND	IM	Entropy	Oracle
AAD	61.73	53.94	1.82	60.19	1.80	1.73	45.42	1.79	59.99	2.08	59.30	1.86	65.69
NRC	57.66	39.94	6.51	62.73	6.86	2.34	21.48	1.58	59.74	1.34	51.54	35.40	64.93
SHOT	59.20	60.90	61.09	60.99	61.09	55.67	56.24	36.46	59.85	37.81	59.79	60.59	64.04
Avg.	59.53	51.59	23.14	61.30	23.25	19.91	41.05	13.28	59.86	13.74	56.88	32.61	64.89
Avg. Rank	4.00	4.67	6.50	2.00	6.50	10.67	7.33	11.33	3.67	10.00	5.00	6.33	-
Source-only													58.03

Table 7. Test-Time Adaptation on CIFAR-10-C. We use the episodic setup where the model is reset after each batch. Only the target domain is available and it is used both for adaptation, validation and computing the oracle performance.

	RankMe	AMI	ARI	V-Measure	FMI	Silhouette	DBI	CHI	BNM	SND	IM	Entropy	Oracle
TENT	83.91	84.01	84.06	84.01	84.06	83.85	28.28	50.68	83.85	20.31	83.85	58.96	84.19
TTT++	80.16	80.05	79.84	80.05	79.71	79.90	59.77	67.29	80.00	58.83	78.98	68.75	80.65
Avg.	82.03	82.03	81.95	82.03	81.89	81.88	44.02	58.98	81.93	39.57	81.42	63.85	82.42
Avg. Rank	3.00	3.00	3.75	3.00	4.25	6.00	11.00	10.00	5.50	12.00	7.50	9.00	-
Source-only													69.71

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5

[7] Ying Jin, Ximei Wang, Mingsheng Long, and Jianmin Wang. Minimum Class Confusion for Versatile Domain Adaptation. In *ECCV*, volume 12366 LNCS, 2020. 5

[8] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International Conference on Machine Learning*, 2020. 2, 5, 7

[9] Jian Liang, Dapeng Hu, and Jiashi Feng. Domain Adaptation with Auxiliary Target Domain-Oriented Classifier. In *CVPR*, 2021. 5

[10] Mingsheng Long, Yue Cao, Jianmin Wang, Michael I Jordan, and Jordan@berkeley Edu. Learning Transferable Features with Deep Adaptation Networks. *arXiv*, 2015. 1, 2, 5

[11] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. Benchmarking validation methods for unsupervised domain adaptation. *arXiv preprint arXiv:2208.07360*, 2022. 3, 5, 6

[12] Kevin Musgrave, Cornell Tech, Serge Belongie, Ser-Nam Lim, and Meta Ai. Unsupervised Domain Adaptation: A Reality Check. In *ECCV*, 2020. 1, 2, 3, 4, 6

[13] V. Patel, R. Gopalan, R. Li, and R. Chellappa. Visual domain adaptation: A survey of recent advances. *Signal Processing Magazine, IEEE*, 32(3):53–69, May 2015. 2

[14] Xingchao Peng, Ben Usman, Neela Kaushik, Dequan Wang, Judy Hoffman, and Kate Saenko. Visda: A synthetic-to-real benchmark for visual domain adaptation. In *ICCV Workshops*, 2018. 2

[15] Luca Robbiano, Muhammad Rameez Ur Rahman, Fabio Galasso, Barbara Caputo, and Fabio Maria Carlucci. Adversarial branch architecture search for unsupervised domain adaptation. In *WACV*, 2022. 2, 5

[16] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Conference on Empirical Methods in Natural Language Processing*, 2007. 4

[17] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, volume 6314 LNCS, 2010. 5

[18] Kuniaki Saito, Donghyun Kim, Piotr Teterwak, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Tune it the right way: Unsupervised validation of domain adaptation via soft neighborhood density. In *ICCV*, 2021. 1, 2, 5

[19] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum Classifier Discrepancy for Unsupervised Domain Adaptation. In *CVPR*, 2018. 5

[20] Yuan Shi and Sha Fei. Information-Theoretical Learning of Discriminative Clusters for Unsupervised Domain Adaptation. In *ICML*, 2012. 2, 4, 5

[21] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *ECCV 2016 Workshops*, 2016. 2, 3, 4, 5

[22] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *ICML*, 2020. 2, 5

[23] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. In *CVPR*, 2014. 2, 3, 4

[24] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. *CoRR*, 2017. 5

[25] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021. 2, 5

[26] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno A. Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *ICLR*, 2021. 2, 3, 7

[27] Shiqi Yang, Joost van de Weijer, Luis Herranz, Shangling Jui, et al. Exploiting the intrinsic neighborhood structure for source-free domain adaptation. *Advances in neural information processing systems*, 34:29393–29405, 2021. 5, 7

[28] Shiqi Yang, Yaxing Wang, Kai Wang, Shangling Jui, et al. Attracting and dispersing: A simple approach for source-free domain adaptation. In *Advances in Neural Information Processing Systems*, 2022. 2, 5, 7

[29] Kaichao You, Ximei Wang, Mingsheng Long, and Michael Jordan. Towards accurate model selection in deep unsupervised domain adaptation. In *ICML*, 2019. 1, 2

[30] Bastien van Delft Baptiste Bellot-Gurlet Taylor Mordan Alexandre Alahi Yuejiang Liu, Parth Kothari. Ttt++: When does self-supervised test-time training fail or thrive? In *NeurIPS*, 2021. 2, 7